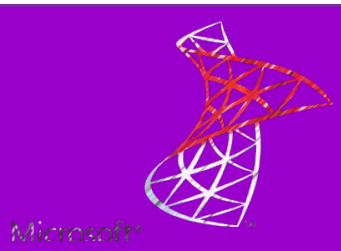


عنوان مقاله: آشنایی با Temporary Table

نویسنده مقاله: غلامحسین عبادی

تاریخ انتشار: خرداد ماه ۱۴۰۱

منبع: https://nikamooz.com/temporary-table/

در تمام زبان‌های برنامه نویسی دنیا ما مفهومی به نام Array یا آرایه را داریم. آرایه یک Data Structure یا ساختمان داده بوده که به ما اجازه می‌دهد تعداد مشخصی مقادیر از یک جنس را درون یک لیست (که ممکن است یک بعدی یا دو بعدی یا n بعدی باشد) نگه داریم و سپس این مقادیر را یکی یکی بخوانیم و یا بنویسیم. به عنوان مثال یک حلقه for بنویسیم که از خانه اول آرایه تا خانه آخر آرایه یکی یکی بخواند و پردازش کند، ولی متاسفانه ما در زبان TSQLآرایه ندارم ولی ما می توانیم با استفاده از جداول Temp یا Temporary Table یا Temp Table ، آرایه‌ها را شبیه‌سازی کنیم. جداول موقت کار آرایه ها را انجام می دهند. پس بنابراین در SQL Server همانطور که گفتیم Temporary Tableیا اصطلاحا (Temp Table کار همان آرایه در سایر زبان‌های برنامه‌نویسی را انجام می‌دهند (ما در زبانTSQL، آرایه نداریم).برای ساخت جداول موقت نیازی به ساخت دیتابیس نداریم (چون این جداول موقت در دیتابیس سیستمی Tempdb ساخته می شوند) و نیاز به هیچگونه دسترسی خاصی نداریم، فقط کافیست آن کاربر اجازه ورود به SQL Server را داشته باشد.

جداول موقت مانند جداول معمولی بوده ولی با این تفاوت که موقت می‌باشند به عبارتی عمر این نوع جداول کوتاه می‌باشد. پس این جداول به چه درد می‌خورد؟ یه زمانی شما در برنامه خود می‌خواهید مقادیری را به طور موقت در جایی نگهداری کنید، خب در سایر زبان‌ها ، برای این کار آرایه می‌سازیم، اینجا آرایه نداریم و به جای آن جدول Tempمی‌سازیم.

پس هر وقت خواستید مقادیر یا اطلاعاتی را به صورت موقت جایی ذخیره کنید، چرا که می‌خواهید از این اطلاعات موقت، در جای دیگری از برنامه استفاده کنید، باید از جداول Temp استفاده کنیم.

انواع جدول Temp

نوع اول جدول Temp با استفاده از دستور Create Table ساخته می‌شود. در این نوع جداول Temp ما قبل از نام جدول از علامت # (نام این کاراکتر شارپ و یا کلید مربع و یا Number Sign یا علامت هشتک) استفاده می‌نماییم.(به این نوع جداول Local Temp Table گفته می‌شود ولی اگر از دو علامت # قبل از نام جدول استفاده کنیم به آن جدول Global Temp Table گفته می شود)

۱. نوع اول جدول Temp با استفاده از دستور Create Table ساخته می‌شود. در این نوع جداول Temp ما قبل از نام جدول از علامت # (نام این کاراکتر شـــارپ و یا کلید مربع و یا Number Sign یا علامت هشـــتک) اسـتفاده می نماییم. (به این نوع جداول Local Temp Table گفته می‌شـود ولی اگر از دو علامت # قبل از نام جدول استفاده کنیم به آن جدول Global Temp Table گفته می‌شود).به شکل زیر توجه کنید.

Syntax

The following syntax describes how to declare a table variable:

```
DECLARE @LOCAL_TABLEVARIABLE TABLE
(column_1 DATATYPE,
 column_2 DATATYPE,
 column_N DATATYPE
)
```

تذکر: ما می‌توانیم برای جدول موقت خود همچون جداول عادی هر چیزی را حتی) Check Constraint جهت کنترل داده های ورودی ، به عبارتی جلو اشتباه کاربر را بگیریم) تعریف نماییم و یا Relation هم تعریف کنیم. (همان رفتاری که با جداول عادی داریم می‌توانیم با جداول موقت هم داشته باشیم.)

۲. نوع دوم جدول Temp با دستور Declare ساخته می‌شود. بعد از کلمه Declare یک نام مناسب به عنوان نام متغیر که قبل از آن از یک کاراکتر @ اسـتفاده شـده (دقت کنید که جنس متغیر ما از نوع Table می‌باشـد) و بعد از نام متغیر از کلمه کلیدی Table اسـتفاده کرده و سـپس فیلدهای جدولمان را مطابق سـاختار زیر ایجاد می نماییم.

Syntax

The following syntax describes how to declare a table variable:

```
DECLARE @LOCAL_TABLEVARIABLE TABLE
(column_1 DATATYPE,
 column_2 DATATYPE,
 column_N DATATYPE
)
```

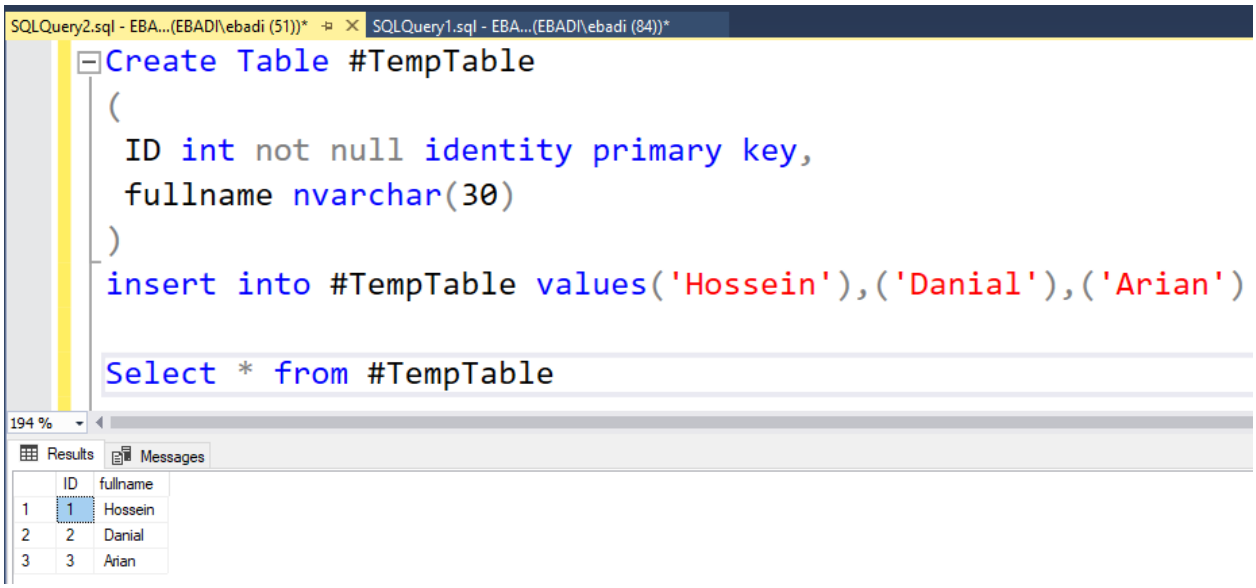
مثال: با استفاده از یک متغیر جدولی نام روزهای هفته و مخفف روزهای هفته را نمایش دهید.

```
DECLARE @ListOWeekDays TABLE
(
DyNumber INT,
DayAbb VARCHAR(40) ,
WeekName VARCHAR(40)
)
INSERT INTO @ListOWeekDays
VALUES
(1,'Mon','Monday')  ,
(2,'Tue','Tuesday') ,
(3,'Wed','Wednesday') ,
(4,'Thu','Thursday'),
(5,'Fri','Friday'),
(6,'Sat','Saturday'),
(7,'Sun','Sunday')
SELECT * FROM @ListOWeekDays
```

مثال: به یک مثال در رابطه با جدول موقت (Temporary Table) دقت کنید.

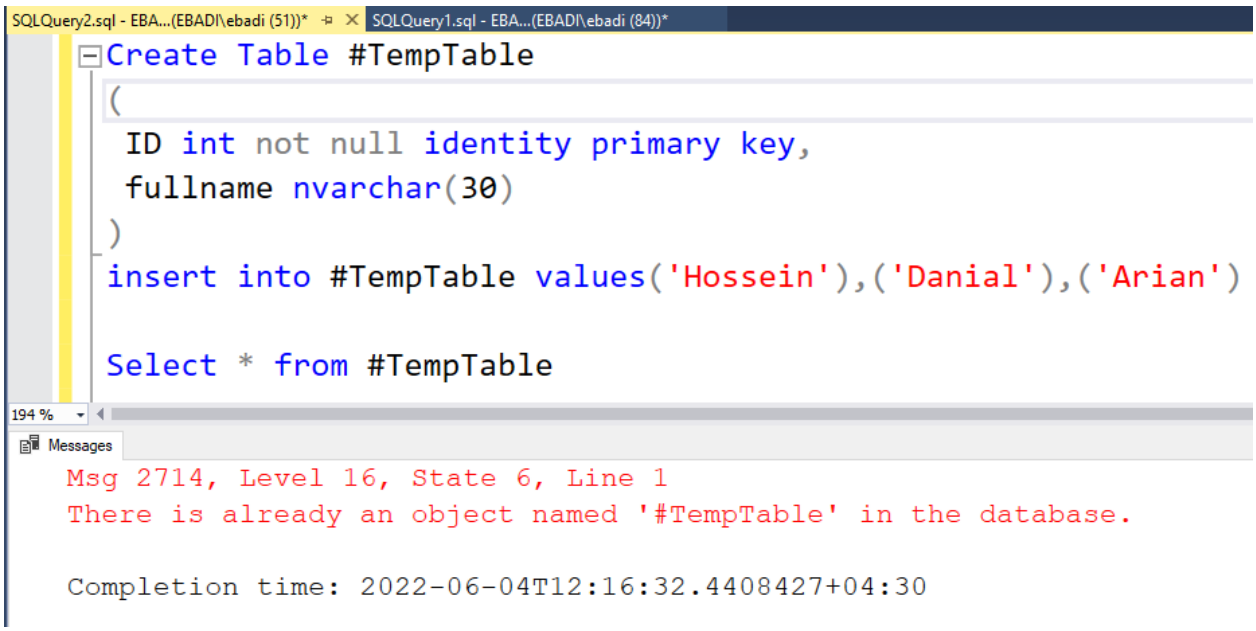
```
Create Table #TempTable
(
 ID int not null identity primary key,
 fullname nvarchar(۳۰)
)
insert into #TempTable values('Hossein'),('Danial'),('Arian')
Select * from #TempTable
```

به خروجی کد فوق در شکل زیر دقت کنید.

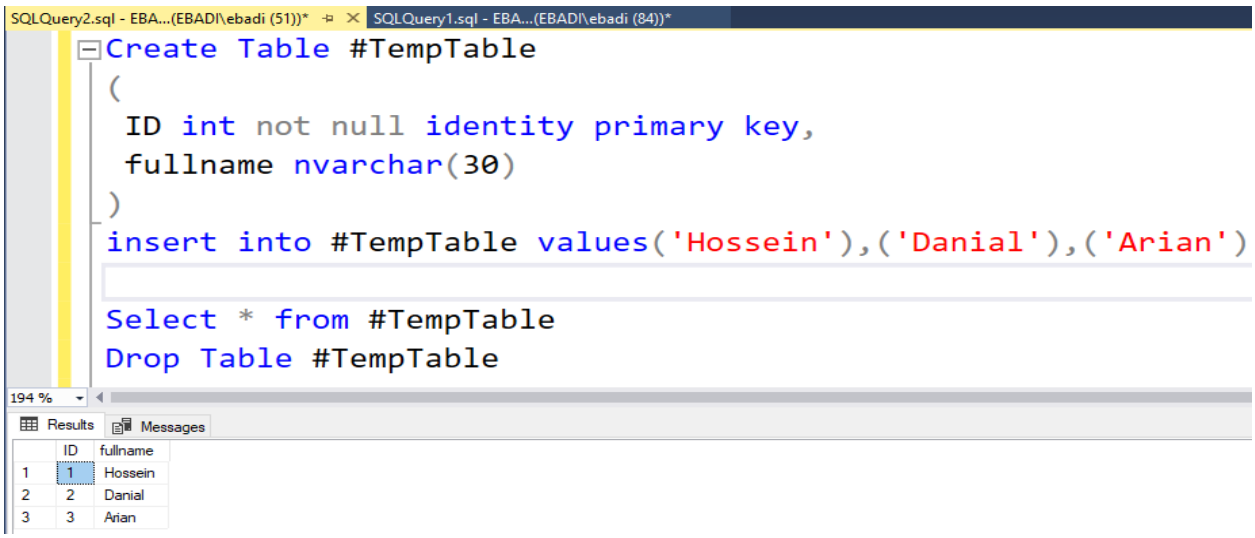


کد فوق را اگر برای بار اول اجر کنید مشــکلی پیش نمی‌آید و با موفقیت اجرا می‌شــود. ولی اگر من دوباره کد فوق را اجرا کنم مطابق شکل زیر با خطا مواجه خواهم شد. این خطا به این معناست که چنین آبجکتی وجود دارد.

عمر این نوع جداول موقت تا وقتی است که یا این جداول را Drop کنیم و یا اینکه آن Session را ببندیم (به عبارتی Disconnect کنیم)

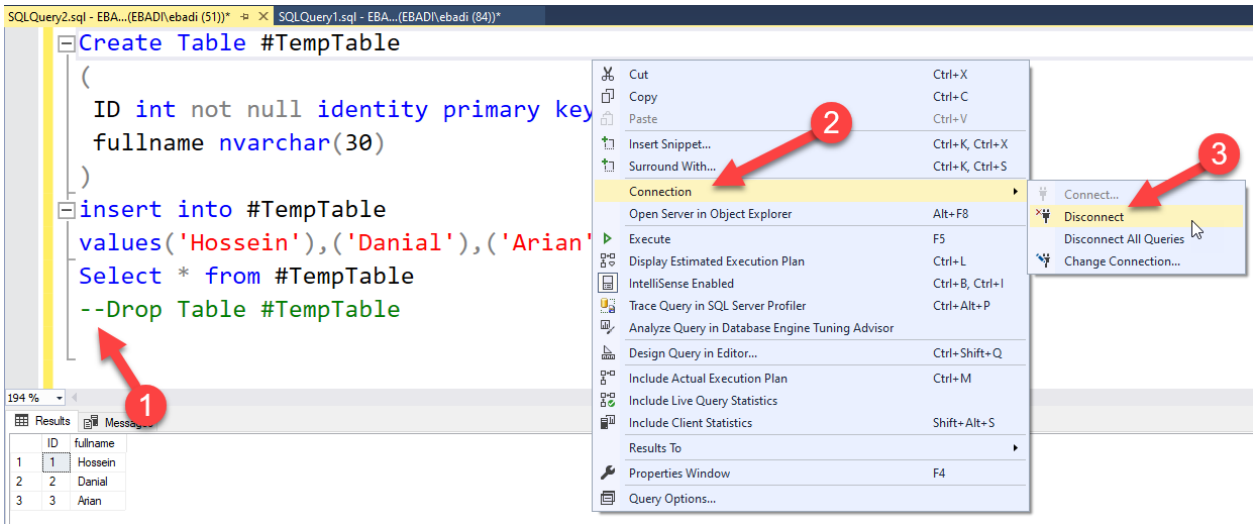


تذکر: بنابراین شما در انتهای کد فوق دستور Drop Table را اضافه نموده و سپس شما ابتدا دستور Drop Table #TempTable اجرا کنید و بعد از این هر چند بار که خواستید کل دستورات را یک جا اجرا کنید و دیگر با خطا مواجه نخواهید شد، چون هر سری داریم جدول موقت را Drop می‌کنیم.

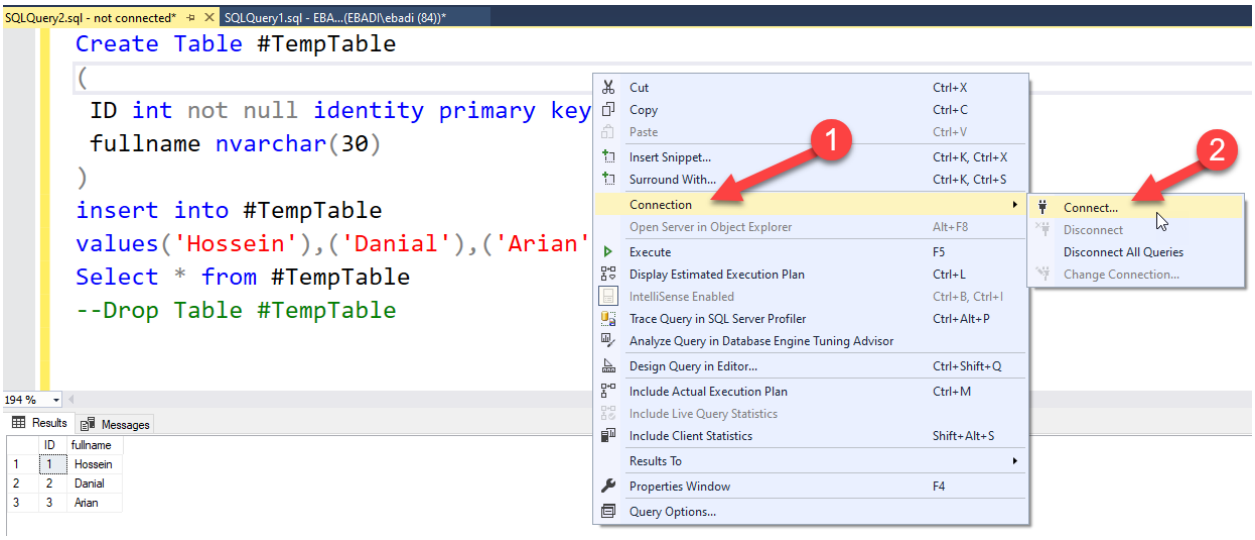


دقت کنید که من می‌توانستم به جای اینکه جدول موقت خود را Drop کنم، می‌توانستم برای حل مشکل Session خود و به عبارتی Connection خود را ببندم. به محض این که Connection بسته شود، جدول Temp از بین می‌رود. حال دوباره کانکشن را ایجاد کرده و دوباره کد را اجرا کنید خواهید دید که خطایی نخواهید گرفت.

خب کافیست دستور Drop Table #TempTable را کامنت کرده و این Session را Disconnect کنید.



حال اگر دوباره کانکشن را بر قرار کرده و برنامه را اجرا کنید به مشکل نخواهید خورد.



تذکر: چرا به این گونه از جداول، جداول موقت گفته می‌شود ؟ برای اینکه به محض بسته شدن کانکشن ، جداول موقت شما از بین می‌روند.

سوال: چرا من باید جدول موقت بسازم؟ چرا ؟ مگر من نمی‌توانم جدول عادی ساخته و بعد از اتمام عملیات، خود آن را پاک کنم؟

به چند دلیل ما اقدام به ساخت جداول موقت می‌نماییم:

- جداول Temp در واقع Private هستند برای شخصی که آن را می سازد. برای اثبات این موضوع کافیست کد فوق را به یک Session جدید کپی کنید و سپس به جای اسامی Hossein و Danial و Arian در Session جدید از نام های دیگر مثلا Mohamad و Esmaeil و Reza استفاده کنید.قبل از شروع به آزمایش بهتر است ابتدا در Session اول دستور Drop Table #TempTable را اجرا کنید. سپس کل دستورات Session اول را اجرا کنید در این صورت اسامی Hossein و Danial و Arian را خواهید دید و اگر دستورات Session دوم را اجرا کنید اسامی Mohamad و Esmaeil و Reza را خواهید دید.

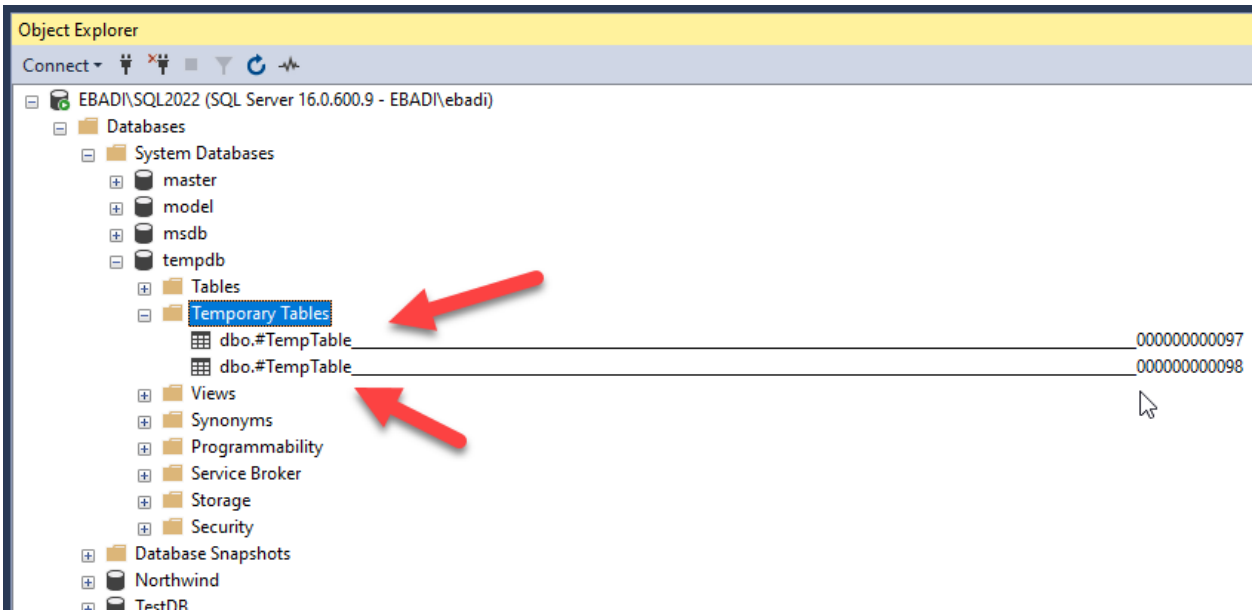
جالبه نه! : با اینکه نام جدول در هر دو Session یکی است، ولی هر کسی Data ی خودش را می‌بیند. مگر چنین چیزی ممکن است! شما به هیچ وجه نمی‌توانید با جدول معمولی چنین کاری را انجام دهید.فرض کنید شما یک Stored Procedure دارید که دارای یک پارامتر وروی سال مالی می‌باشد.کاربر اول این SP را برای سال مالی ۱۴۰۰ صدا می زند و یک جدول موقتی ساخته شده و اطلاعات سال ۱۴۰۰ در آن ریخته می شود و کاربر دومی به طور همزمان این SP را برای سالی مالی ۱۴۰۱ صدا می‌زند و برای او هم یک جدول موقت ساخته شده و اطلاعات سال ۱۴۰۱ در آن ریخته می‌شود. این دیتاها با هم قاطی نمی‌شوند. کاربر یک داده های خود را می بیند و کاربر دوم نیز داده های خود را می‌بیند.اگر بنا بود از جداول موقت استفاده نکنید باید نام جدول و نام کاربر و تاریخ و زمان را به هم ملحق می‌کردید تا یک نام یونیک ایجاد کرده و نهایتا باید

به سمت Dynamic Query می رفتید. بنابراین چون نام جدول را باید متغیر بسازید باید به سمت Dynamic Queryبروید. بنابراین ما مطمئن هستیم که جداول موقت ما با جداول موقت دیگر قاطی نمی‌شوند.

تذکر: تا اینجا متوجه شدیم که حسن جدول موقت چی بود ؟ جداول موقت Local یا Private می‌باشند. به عبارتی جدول موقت هر کسی مالِ خودش می‌باشد.

ممکن است شما سوال کنید که اسکیوال سرور چگونه این موضوع را مدیریت (Handle) می کند؟ اسکیوال سرور جداول موقت را در دیتابیس جاری نمی‌سازد بلکه آنها را در دیتابیس سیستمی Tempdb می سازد. اگر دیتابیس Tempdb را باز کنیم ، شاخه‌ایی به نام Temporary Table را خواهید دید.

تذکر: اگر شما در هر دو Session دستور Drop Table #TempTable را کامنت کنید و هر دو Session را اجرا کنید در این صورت مطابق شکل زیر خواهید دید که این جداول در دیتابیس Tempdb و در قسمت Temporary Tableمطابق شکل زیر ایجاد می‌شوند.



حال اگر شما یکی ازاین Sessionها را Disconnect کنید و یا اینکه از دستور Drop Table #TempTable استفاده کنید در این صورت یکی از این جداول موقت از دیتابیس Tempdb و از قسمت Temporary Tableحذف خواهند شد.

- حسن دوم جداول موقت همانطور که گفته شد این است که در دیتابیس TempDB ساخته می‌شوند. چون اگر بنا بود کاربری در جدول اصلی جدول بسازد باید به آن فرد حق Create Table می دادیم، در صورتی که در دیتابیس Tempdb همه حق ساختن جداول موقت را دارند(حتی اگر دسترسی یا Permission ساخت جدول را نداشته باشند). به عبارتی هر کسی که بتواند به SQL Server لاگین کند می‌تواند در دیتابیس TempDBجداول موقت بسازد و برنامه نویسان می‌توانند بدون اینکه ما به آنها دسترسی ایجاد جدول موقت بدهید، می توانند در برنامه‌هایشان جداول Temp بسازند و از آنها استفاده کنند.

- در بعضی از موارد می توانید مقادیریی را داخل جداول موقت ریخته و سپس این جدول موقت را با جدول اصلی Join بزنید که در این صورت باعث افزایش سرعت کوئری و افزایش Performance خواهید شد.(البته باید با حالت اول به کمک Execution Plan مقایسه شود)

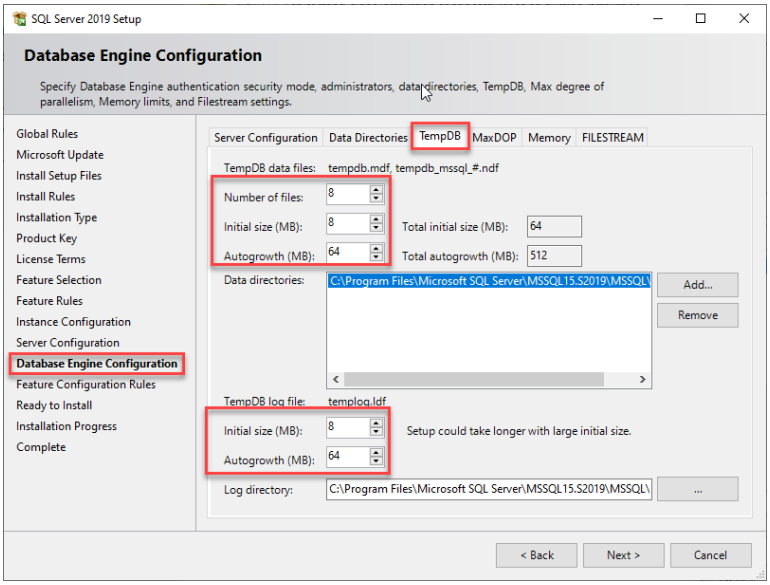
تذکر: اگر در شرکتی کار می کنید که بسیار بزرگ بوده و کاربران زیادی هم زمان مشغول کار هستند و برنامه نویسان هم عادت دارند که جداول موقت می‌سازند، در این صورت فشار خیلی زیادی روی دیتابیس TempDB خواهیم داشت. چرا ؟ چون دائم جداول موقت ایجاد و سپس حذف می شوند. شما در هیچ دیتابیس دیگری این رفتار را نمی بینید که دائما جداولی در آن دیتابیس ساخته شده و سپس پاک شود و فقط TempDB این گونه است. دقت کنید اگر TEMPDB در این گونه شرکت ها Optimize نشده باشد در این صورت ایجاد جداول موقت باعث کندی خواهد شد.

برای Optimize دیتابیس TempDB کافیست فایل دیتای این دیتابیس را به هشت دیتا فایل افزایش دهید. مگر اینکه به کمک مانیتورینگ متوجه شوید که به تعداد دیتا فایل بیشتری نیاز دارد.(مثلا به کمک Red-gate و یا به کمک Apex و یا به کمک Idera و غیره.

تذکر: گفته می‌شود که بهتر است تعداد دیتا فایل های دیتابیس Tempdb باید به اندازه تعداد هسته‌های منطقی یا Logical ، CPU سرور، فایل داشته باشد.

تذکر: Optimize کردن دیتابیس Tempdb به قدر حائز اهمیت بود که مایکروسافت از نسخه SQL Server 2016 تنظیمات مربوط به آن را در هنگام نصب اضافه کرده است. بنابراین از نسخه SQL ۲۰۱۶ نگاه میکنه ببینه که شما چند تا هسته داری ، سپس Tempdb شما رو Optimize طراحی می کند. البته این کار را تا هشت تا Core انجام می‌دهد. اگر به فرض ۱۶ تا Core داشته باشید باز هم به شما هشت تا دیتا فایل پیشنهاد می‌دهد.

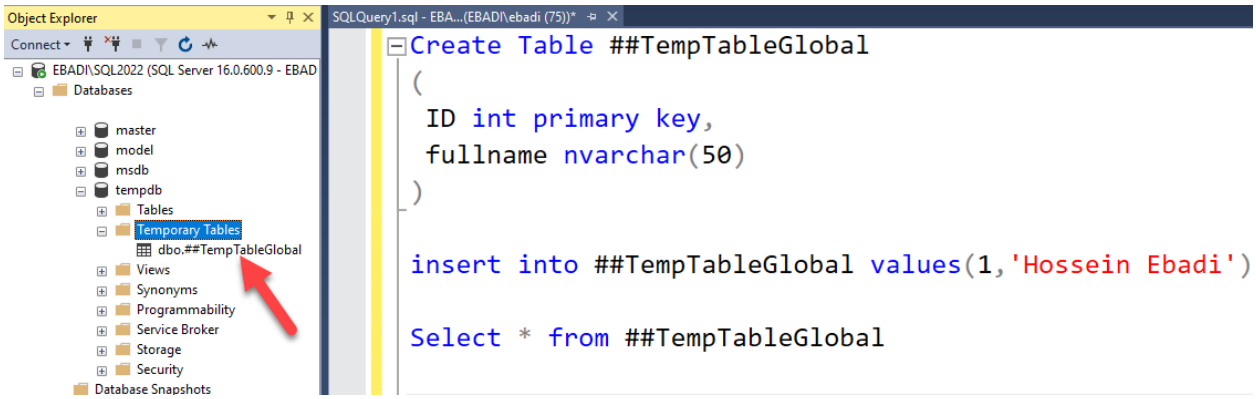
نکات مربوط به افزایش Performance دیتابیس سیستمی TempDB



- دیتا فایل‌های دیتابیس Tempdb خود را نهایتا به هشت فایل افزایش دهید. مگر اینکه مطمئن شوید که واقعا نیاز است که تعداد دیتا فایل‌ها باید افزایش یابد.
- نکته خیلی مهم دیگری که باید رعایت کنید این است که سایز این دیتا فایل‌ها را یکی و نحوه رشد آنها را نیز یکی اختیار کنید.
- برای افزایش Performance بهتر است که از هاردهای پرسرعت (هاردهای SSD) استفاده کنید.

- نکته بعدی که می‌تواند باعث افزایش پرفورمنس دیتابیس TempDB شود این است که هارد مربوط به آن را کاملا و به صورت فیزیکی از سایر هارد جدا کنید.

تذکر مهم: اگر قبل از نام جداول موقت (آنهایی که با Create Table شروع می‌شوند) از یک کاراکتر # استفاده کنیم به آن جدول موقت، جدول موقت Local گفته می‌شود ولی اگر قبل از نام جداول موقت از دو کاراکتر ## استفاده کنیم به آن جدول موقت Global گفته می‌شود. جداول موقت Local در Session فعلی و جداول موقت Global در تمامی Sessionها قابل دسترس و مشاهده هستند. دقت کنید که هر دو در دیتابیس Tempdb ساخته می‌شوند.به شکل زیر دقت کنید. جدول موقت global زیر یعنی ##TempTableGlobal از Sessionهای دیگر نیز قابل دسترس می‌باشد.



جدول موقت نوع دوم یا Table Variable :

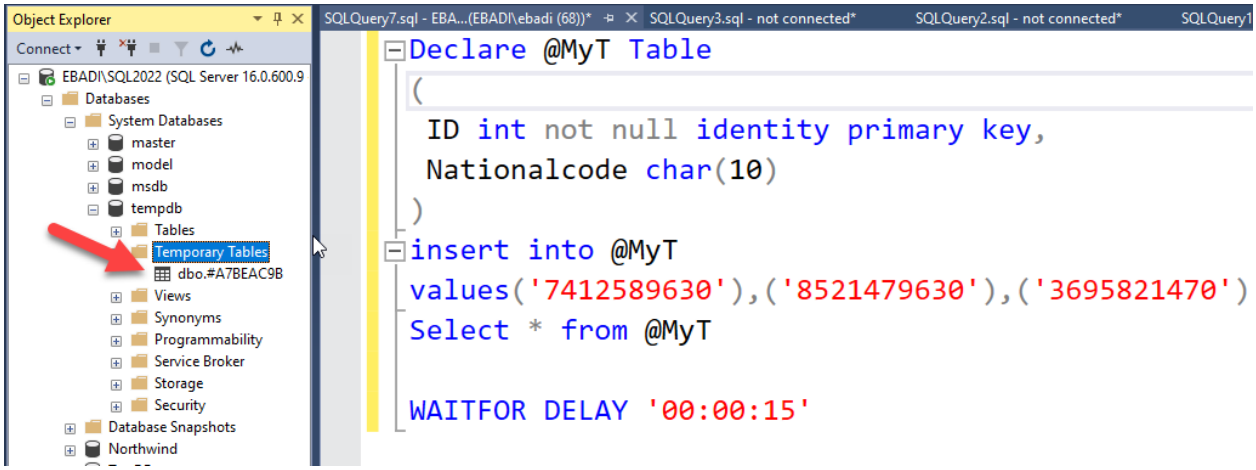
جداولی که از نوع دوم یعنی Table Variable می‌باشند یعنی با Declare تعریف می‌شوند عمرشان مثل متغیرها می‌باشد. یعنی عمر آنها در حد یک Bach می‌باشد و به محض اتمام برنامه آنها نیز از بین می‌روند. لذا اگر این برنامه را ما دوباره اجرا کنیم خطا نخواهیم گرفت (بر خلافت جداول موقت نوع اول). لذا این نوع جداول نیاز به Drop Table ندارند. حتی در جداول موقت نوع دوم اگر شما بخواهید دستور Drop استفاده کنید به شما خطا خواهد داد.

به مثال زیر توجه کنید :

```
Declare @MyT Table
(
 ID int not null identity primary key,
 Nationalcode char(۱۰)
)
insert into @MyT values ('۷۴۱۲۵۸۹۶۳۰'), ('۸۵۲۱۴۷۹۶۳۰'), ('۳۶۹۵۸۲۱۴۷۰')
Select * from @MyT
```

نکته حائز اهمیت این است که Table Variableها (بر خلاف نظر خیلی از دوستان عزیز) نیز در دیتابیس TempDB ساخته می شوند. چگونه ثابت کنم که جداول موقت نوع دوم نیز در Tempdb ساخته می‌شوند. کافیست در انتهای کد فوق دستور WAITFOR DELAY '۰۰:۰۰:۱۵' را اضافه کنیم. همانطور که در شکل زیر می بینید جدول موقت @MyT در Temporary Table ساخته می‌شود.

به کمک دستور WAITFOR DELAY '۰۰:۰۰:۱۵' عملا جدول موقت من پانزده ثانیه دیرتر از بین می‌رود.



Intelligent Query Processing (IQP):

از نسخه SQL Server 2017 یکسری ویژگی‌هایی در جهت کاهش هزینه (Cost) اجرای کوئری‌ها به Engine SQL Server اضافه شدند. این ویژگی‌ها به صورت هوشمندانه و بدون نیاز به تغییر کدهای شما باعث کاهش زمان اجرای کوئری ها و کاهش IO و کاهش مصرف حافظه (Memory) شدند.در واقع هدف از IQP این بود که Developerها و DBA ها به کدهای خود دست نزده و به طور اتوماتیک پرفورمنس کدهای آنها بهبود یابد. یکسری از این تغییرات در SQL Server 2017 ایجاد شده و در ادامه، یکسری از این تغییرات در SQL Server 2019 و در ادامه تغییراتی نیز در SQL Server 2022 ایجاد خواهد شد.

به بعضی از این تغییرات در شکل دقت کنید :

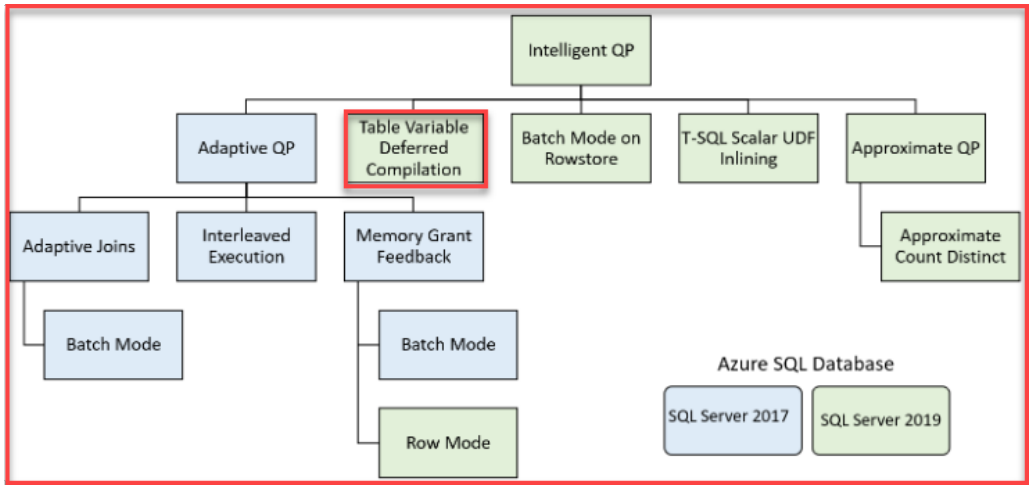
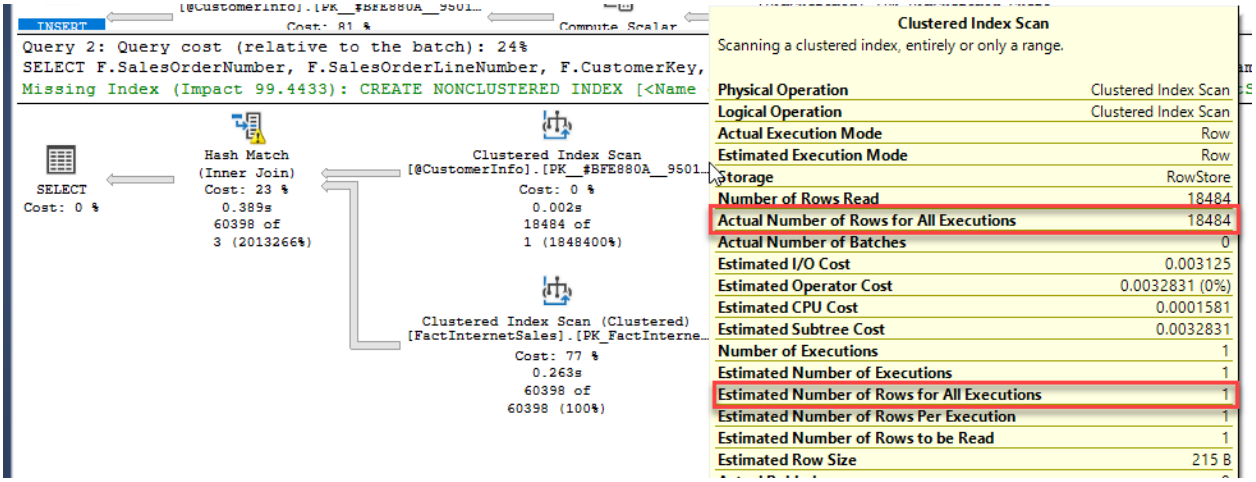


Table Variable Deferred Compilation:

بزرگترین مشکل متغیرهای جدولی (Table Variable) تخمین اشتباه تعداد رکوردهای بازگشتی می باشد (Cardinality Estimation اشتباه است) این تخمین در نسخه‌های قبل از SQL Server 2019 عدد یک می‌باشد. بعضی مواقع شما می بینید که SPهای شما کند می‌باشند که ممکن است یکی از دلایل آن وجود متغیرهای جدولی باشد. ولی خوشبختانه این مشکل در SQL Server 2019 حل شده است البته بدون اینکه شما به کدهای خود دست بزنید. لازم به ذکر است که برنامه‌نویسان و DBAها، این مشکل را در نسخه‌های قبل از SQL Server 2019، به کمک دستور option(recompile) حل می‌کردند.

مطابق شکل زیر اجرا کوئری در نسخه SQL Server 2017 صورت گرفته ولی متاسفانه تخمین زده که با یک رکورد طرف حساب است (Estimate Execution Plan)، ولی متاسفانه در عمل می‌بینه که با 18484 رکورد (Actual Execution Plan) طرف حساب است.



حال Compatibility Level را به SQL Server2019 ارتقاء داده و کد خود را اجرا می‌نماییم، حال به شکل زیر دقت کنید خواهید دید که Estimate Execution Plan و Actual Execution Plan یکی خواهد شد.